Ch. 4 Repetition Structures

Iteration: one execution of the body of a loop

Infinite loop: loop that does not have a way of stopping

Iterable: contains a sequence of values that can be iterated over

Sentinel: special value that marks the end of a sequence of items

eg; `while lot ! = 0:` from program 4-13 in the text

The for Loop: a Count-Controlled Loop

```
for variable in [value1, value2, etc.]:
    statement
    statement
    etc.
```

Iterates a specific number of times (each item in the sequence)

for loops - Range Function

```
for num in range(5):
    print(num)
for num in range(0, 5):
    print(num)
for num in [0, 1, 2, 3, 4]:
    print(num)
```

All three of these execute a For loop 5 times, and print the numbers [0-4]

The range function simplifies the process of writing a for loop, since it returns an iterable object

One argument: used as ending limit

Two arguments: starting value and ending limit

Three arguments: third argument is step value

Can generate a sequence of numbers in *descending order*, if the starting value is larger than the end limit and the step value is negative. Eg;

`range (10, 0 -1)`

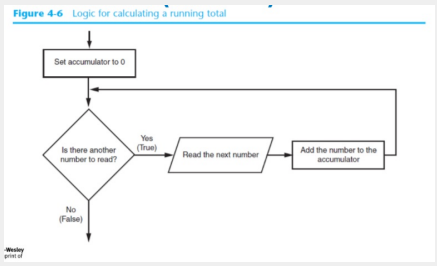
The while Loop: a Condition-Controlled Loop

```
while condition:
            statements
```

"While condition is true, do something. " Also known as a "pretest" loop, since it tests for a condition before performing an iteration. Also used as **Input Validation** loops.

- Condition tested for true or false value
- Statements repeated as long as condition is true

Accumulator Variable



Augmented Assignment Operators

Operator	Example Usage	Equivalent To
+=	x += 5	x = x + 5
-=	y -= 2	y = y - 2
*=	z *= 10	z = z *10
/=	a /= b	a = a / b

Augmented Assignment Operators (cont)

%=	c %= 3	c = c % 3

Shorthand operators

Input Validation Loops

```
# Get a test score.
score = int(input('Enter a test score: '))
# Make sure it is not less than 0.
while score < 0:
    print('ERROR: The score cannot be negative.')
    score = int(input('Enter the correct score: '))
```

Input validation is the process of inspecting data that has been input to a program, to make sure it is valid before it is used in a computation. Input validation is commonly done with a loop that iterates as long as an input variable references bad data.

E.g.; This input validation loop (using a while loop) rejects any input that isn't zero, and asks for new input.

Nested Loop -- Loop contained inside another loop

```
# This program displays a rectangular pattern
   # of asterisks.
   rows = int(input('How many rows? '))
   cols = int(input('How many columns? '))

   for r in range(rows):
       for c in range(cols):
          print('*', end='')
       print()
```

Chapter 4 Questions

4.1 What is a repetition structure?

> A structure that causes a section of code to repeat

4.2 What is a condition-controlled loop?

> A loop that uses a true/false condition to control the number of times that it repeats

4.3 What is a count-controlled loop?

> A loop that repeats a specific number of times

4.4 What is a loop iteration?

> An execution of the statements in the body of the loop

Chapter 4 Questions (cont)

4.5 Does the while loop test its condition before or after it performs an iteration?

> Before

4.6 How many times will 'Hello World' be printed in the following program?

```
count = 10
while count < 1:
    print('Hello World')
```

> None. The condition `count < 0` will be false to begin with.

4.7 What is an infinite loop?

> A loop that has no way of stopping and repeats until the program is interrupted.

4.8 Rewrite the following code so it calls the`range` function instead of using the list `[0, 1, 2, 3, 4, 5]`:

```
for x in [0, 1, 2, 3, 4, 5]:
   print('I love to program!')
```

```
for x in range(6):
   print('I love to program!')
```

4.9 What will the following code display?

```
for number in range(6):
  print(number)
```

> 0
> 1
> 2
> 3
> 4
> 5

4.10 What will the following code display?

```
for number in range(2, 6):
   print(number)
```

> 2
> 3
> 4
> 5

Chapter 4 Questions (cont)

4.11 What will the following code display?

```
for number in range(0, 501, 100):
   print(number)
```

> 0
> 100
> 200
> 300
> 400
> 500`

4.12 What will the following code display?

```
for number in range(10, 5, -1):
   print(number)
```

> 10
> 9
> 8
> 7
> 6

4.13 What is an accumulator?

> A variable that is used to accumulate the total of a series of numbers

4.14 Should an accumulator be initialized to any specific value? Why or why not?

> Yes, it should be initialized with the value 0. This is because values are added to the accumulator by a loop. If the accumulator does not start at the value 0, it will not contain the correct total of the numbers that were added to it when the loop ends.

4.15 What will the following code display?

```
total = 0
for count in range(1, 6):
   total = total + count
print(total)
```

> 15

Chapter 4 Questions (cont)

4.16 What will the following code display?

```
number 1 = 10
number 2 = 5
number 1 = number 1 + number 2
print(number1)
print(number2)
```

```
15
5
```

4.17 Rewrite the following statements using augmented assignment operators:

a. `quantity = quantity + 1`

b. `days_left = days_left - 5`

c. `price = price * 10`

d. `price = price / 2`

a. `quantity += 1`

b. `days_left -= 5`

c. `price *= 10`

d. `price /= 2`

4.18 What is a sentinel?

> A sentinel is a special value that marks the end of a list of items.

4.19 Why should you take care to choose a distinctive value as a sentinel?

> A sentinel value must be unique enough that it will not be mistaken as a regular value in the list.

4.20 What does the phrase “garbage in, garbage out” mean?

> It means that if bad data (garbage) is provided as input to a program, the program will produce bad data (garbage) as output.

Chapter 4 Questions (cont)

4.21 Give a general description of the input validation process.

> When input is given to a program, it should be inspected before it is processed. If the input is invalid, then it should be discarded and the user should be prompted to enter the correct data.

4.22 Describe the steps that are generally taken when an input validation loop is used to validate data.

> The input is read, then a pretest loop is executed. If the input data is invalid, the body of the loop executes. In the body of the loop, an error message is displayed so the user will know that the input was invalid, and then the input read again. The loop repeats as long as the input is invalid.

4.23 What is a priming read? What is its purpose?

> It is the input operation that takes place just before an input validation loop. The purpose of the priming read is to get the first input value.

4.24 If the input that is read by the priming read is valid, how many times will the input validation loop iterate?

> None

Ch 5 Functions

Function: group of statements within a program that perform as specific task

Modularized program: program wherein each task within the program is in its own function

Void Function: Simply executes the statements it contains and then terminates.

A value-returning function: Executes the statements it contains, and then it returns a value back to the statement that called it.

The input, int, and float functions are examples of value-returning functions.

Function definition: specifies what function does

```
def function_name():
        statement
        statement
```

Function header: first line of function. Includes keyword `def` and function name, followed by parentheses and colon

Block: set of statements that belong together as a group

Example: the statements included in a function

main function: called when the program starts

Chapter 5 Questions

5.1 What is a function?

> A function is a group of statements that exist within a program for the purpose of performing a specific task.

5.2 What is meant by the phrase “divide and conquer”?

> A large task is divided into several smaller tasks that are easily performed.

5.3 How do functions help you reuse code in a program?

> If a specific operation is performed in several places in a program, a function can be written once to perform that operation, and then be executed any time it is needed.

5.4 How can functions make the development of multiple programs faster?

> Functions can be written for the common tasks that are needed by the different programs. Those functions can then be incorporated into each program that needs them.

5.5 How can functions make it easier for programs to be developed by teams of programmers?

> When a program is developed as a set of functions in which each performs an individual task, then different programmers can be assigned the job of writing different functions.

Chapter 5 Questions (cont)

5.6 A function definition has what two parts?

A function definition has two parts: a header and a block. The header indicates the starting point of the function, and the block is a list of statements that belong to the function.

5.7 What does the phrase “calling a function” mean?

To call a function means to execute the function.

5.8 When a function is executing, what happens when the end of the function’s block is reached?

When the end of the function is reached, the computer returns back to the part of the program that called the function, and the program resumes execution at that point.

5.9 Why must you indent the statements in a block?

Because the Python interpreter uses the indentation to determine where a block begins and ends

5.10 What is a local variable? How is access to a local variable restricted?

A local variable is a variable that is declared inside a function. It belongs to the function in which it is declared, and only statements in the same function can access it.

5.11 What is a variable’s scope?

The part of a program in which a variable may be accessed

Chapter 5 Questions (cont)

5.12 Is it permissible for a local variable in one function to have the same name as a local variable in a different function?

Yes, it is permissible.

5.13 What are the pieces of data that are passed into a function called?

Arguments

5.14 What are the variables that receive pieces of data in a function called?

Parameters

5.15 What is a parameter variable’s scope?

A parameter variable’s scope is the entire function in which the parameter is declared.

5.16 When a parameter is changed, does this affect the argument that was passed into the parameter?

No, it does not.

5.17 The following statements call a function named `show_data`. Which of the statements passes arguments by position, and which passes keyword arguments?

a. `show_data(name='Kathryn', age=25)`

b. `show_data('Kathryn', 25)`

passes by keyword argument passes by position

Chapter 5 Questions (cont)

5.18 What is the scope of a global variable?

> The entire program

5.19 Give one good reason that you should not use global variables in a program.

> Here are three:
>
> Global variables make debugging difficult. Any statement in a program can change the value of a global variable. If you find that the wrong value is being stored in a global variable, you have to track down every statement that accesses it to determine where the bad value is coming from. In a program with thousands of lines of code, this can be difficult.
>
> Functions that use global variables are usually dependent on those variables. If you want to use such a function in a different program, you will most likely have to redesign it so it does not rely on the global variable.
>
> Global variables make a program hard to understand. A global variable can be modified by any statement in the program. If you are to understand any part of the program that uses a global variable, you have to be aware of all the other parts of the program that access the global variable.

Chapter 5 Questions (cont)

5.20 What is a global constant? Is it permissible to use global constants in a program?

> A global constant is a name that is available to every function in the program. It is permissible to use global constants. Because their value cannot be changed during the program's execution, you do not have to worry about its value being altered.

5.21 How does a value-returning function differ from the void functions?

> The difference is that a value returning function returns a value back to the statement that called it. A simple function does not return a value.

5.22 What is a library function?

> A prewritten function that performs some commonly needed task

5.23 Why are library functions like "black boxes"?

> The term "black box" is used to describe any mechanism that accepts input, performs some operation (that cannot be seen) using the input, and produces output.

5.24 What does the following statement do? `x = random.randint(1, 100)`

> It assigns a random integer in the range of 1 through 100 to the variable `x`.

Chapter 5 Questions (cont)

5.25 What does the following statement do?

```
print(random.randint(1, 20))
```

> It prints a random integer in the range of 1 through 20.

5.26 What does the following statement do?

```
print(random.randrange(10, 20))
```

> It prints a random integer in the range of 10 through 19.

5.27 What does the following statement do?

```
print(random.random())
```

> It prints a random floating-point number in the range of 0.0 up to, but not including, 1.0.

5.28 What does the following statement do?

```
print(random.uniform(0.1, 0.5))
```

> It prints a random floating-point number in the range of 0.1 through 0.5.

5.29 When the `random` module is imported, what does it use as a seed value for random number generation?

> It uses the system time, retrieved from the computer's internal clock.

5.30 What happens if the same seed value is always used for generating random numbers?

> If the same seed value were always used, the random number functions would always generate the same series of pseudorandom numbers.

Chapter 5 Questions (cont)

5.31 What is the purpose of the `return` statement in a function?

> It returns a value back to the part of the program that called it.

5.32 Look at the following function definition:

```
def do_something(number):
   return number * 2
```

a. What is the name of the function?
b. What does the function do?
c. Given the function definition, what will the following statement display?

```
print(do_something(10))
```

> a. `do_something`
> b. It returns a value that is twice the argument passed to it.
> c. 20

5.33 What is a Boolean function?

> A function that returns either `True` or `False`

5.34 What `import` statement do you need to write in a program that uses the `math` module?

```
import math
```

5.35 Write a statement that uses a`math` module function to get the square root of 100 and assigns it to a variable.

```
square_root = math.sqrt(100)
```

5.36 Write a statement that uses a`math` module function to convert 45 degrees to radians and assigns the value to a variable.

```
angle = math.radians(45)
```